Qt6课程设计软件文档

——OICQ



小 组：第16组

组 员：吴博渊、邓鑫 、黄秋汕

课程名称：Qt6课程设计

学科专业：22级软件工程2班

所在学院：计算机与信息科学学院

重庆师范大学

2024年7月8日

**目 录**

1. **开发阶段**
2. **需求规格说明书**
3. **功能需求**
4. **用户界面需求**
5. **概要设计说明书**
6. **项目概述**
7. **系统架构**
8. **功能模块设计**
9. **数据库设计**
10. **技术选型**
11. **开发计划**
12. **项目团队与分工**
13. **开发环境与工具**
14. **功能模块设计**
15. **数据库设计**
16. **开发计划**
17. **实施阶段**
18. **开发进度**
19. **代码审查**
20. **问题跟踪**
21. **用户文档**
22. **测试计划**
23. **测试实施**
24. **关键技术及难点**
25. **开发心得与体会**

**一、开发阶段**

1. 需求规格说明书
2. 功能需求

· **用户登录**

* 用户输入账号密码登录（判断密码错误、重复登陆、联网失败等情况）

· **好友管理**

* 添加好友：支持通过搜索用户名方式添加好友，显示添加好友列表。
* 好友列表：显示好友头像、昵称等信息，能预览显示最新消息和未读消息。

· **即时通讯**

* 文本消息：支持发送文字消息，支持表情、自定义贴纸等富文本内容。
* 文件消息：支持上传文件、下载文件，如文档、视频等。
* 屏幕截图：支持选择不同的模式截取屏幕并插入到聊天框中。
* 图片浏览器：支持查阅聊天过程中已发送的图片，具有前后翻阅、缩放、旋转等浏览方式。
* 离线消息：支持离线消息接收。
* **个性化设置**
* 头像上传与修改：支持自定义头像。

1. 用户界面需求
2. 界面设计原则
3. **一致性**：保持界面元素和交互模式的一致性，减少用户学习成本。
4. **简洁性**：界面设计应简洁明了，避免过多的视觉干扰，确保信息有效传达。
5. **可用性**：界面布局合理，操作流程顺畅，确保用户能够高效完成任务。
6. **美观性**：符合现代审美，运用色彩、图标、字体等元素提升视觉吸引力。
7. **可访问性**：考虑不同用户群体的需求，如色弱用户、残障人士等，确保界面易于访问和使用。
8. 主界面设计
9. **登录界面**

设计简洁的登录表单，包含用户名、密码输入框及登录按钮。

1. **主界面布局**
   1. 采用侧边栏或底部导航栏设计，便于用户快速切换聊天、联系人、动态、设置等功能模块（部分功能尚未实现）。
   2. 聊天列表显示最近联系人及未读消息状态信息。
   3. 顶部状态栏用户头像及昵称等信息。
2. **聊天窗口**
   1. 聊天窗口包含输入框、发送按钮、表情/贴纸、截图、文件选择按钮等。
   2. 支持文本、视频、图片、文件等多种消息类型的发送与接收。
   3. 消息列表显示时间戳、消息内容、未读消息条数等操作。
3. **个性化设置**

提供丰富的个性化设置选项，如头像等。

1. 视觉设计

* 色彩搭配应和谐统一，符合用户群体的审美偏好。
* 图标设计应简洁明了，易于识别和理解。
* 字体选择应清晰易读，大小适中。
* 布局设计应合理有序，避免视觉拥挤和混乱。

1. 交互设计

* 确保所有交互操作都符合用户直觉，减少不必要的点击和滑动次数。
* 提供清晰的反馈机制，如操作成功/失败提示、加载动画等，提升用户体验。
* 支持鼠标滚轮等现代交互方式，提高操作效率。

1. 概要设计说明书
2. 项目概述

本项目旨在开发一款仿照腾讯QQ的即时通讯软件，名为“OICQ”，旨在为用户提供高效、便捷、安全的在线交流体验。通过文字、图片、视频等多种沟通方式，满足用户在社交、工作、学习等多场景下的即时通讯需求。同时，融入个性化设置、多样化使用、文件传输等丰富功能，提升用户体验，打造一个综合性的社交平台。

1. 系统架构

* 客户端架构
* **UI层**：使用QML，负责用户界面的渲染和交互逻辑的处理。
* **Controller层**：处理用户请求，进行数据校验，调度model，网络套接字等等对象。
* **Model:** 包括FriendModel,MessageModel,FriendRequestModel等等
* **网络交互的Socket**: 负责网络发送数据
* 服务端架构

**网络交互的Socket:** 负责接收数据

**网络Server层**：处理转发，进行数据管理

**实体层:** 将数据库中常见的数据映射到内存中进行管理。

**数据层**：存储数据到数据库

1. 功能模块设计

· **用户管理模块**

* 登录：登录以管理用户数据。
* 个人信息管理：允许用户编辑头像。
* 好友管理：处理其他用户的好友添加请求，发起好友添加请求。

· **即时通讯模块**

* 文本消息：支持发送和接收文字、图片、表情，混合文字和图片内容。
* 未读消息：红点提醒。
* 图片消息的浏览：双击图片消息（边框）可打开图片浏览器查看当前聊天窗口中的图片，支持图片放大、缩小、拖动、旋转。

· **文件传输模块**

* 支持在聊天窗口内发送和接收文件，支持最多同时发送5个文件等。
* 发送文件途中再次点击文件可取消发送，发送途中显示发送进度。
* 接收方和发送放都支持文件下载。

1. 数据库设计

· **用户表**：存储用户的基本信息，如QQ号、密码、昵称、头像等。

· **好友表**：存储用户与好友的关系信息，以及状态信息

· **聊天记录表**：存储用户之间的聊天记录。

· **好友请求**表****：存储添加好友的所有请求。

· **文件映射**表****：存储文件和消息之间的映射。

1. 开发计划
2. 项目团队与分工

吴博渊（组长）：负责整体的架构设计，c++与qml交互，c++业务逻辑实现(网络传输和Model)，远程仓库的管理及维护，初版qml的页面。

邓鑫：负责截图、输入框复制粘贴，聊天气泡响应式的等功能实现，聊天界面、登录界面以及部分页面的重构及优化，对于错误显示的界面的实现，软件文档的编写。

黄秋汕：负责表情包界面和发送、调整窗口子项大小，好友添加窗口，窗口侧边栏及逻辑实现，组件的重构及优化，组件部分逻辑的优化，演示视频的剪辑。

1. 开发环境与工具

* **开发环境**：QT Manjaro KDE
* **编程语言**：QML/C++
* **数据库**：sqlite数据库

**二、实施阶段**

1. 问题跟踪

**问题记录**：如发现开发过程中出现错误，则将错误记录下来，等待处理。

**分配任务**：项目组长将问题分配给相应的开发人员进行处理。

**问题解决**：检查是否能够编译运行，对不规范(格式，没必要的复杂)的代码进行修改，对多于的文件和资源进行删除。

**三、用户文档**

1. 用户手册

**一、软件简介**

OICQ是一款功能丰富、界面友好的即时通讯软件，支持文字、图片、表情聊天，文件传输等多种功能。

**二、启动与登录**

1. ****启动软件：从github克隆仓库，运行shell文件进行部署(初始化数据库，创建桌面图标等等)。然后命令行启动server，桌面图标或应用菜单点击启动client。****
2. ****或者使用源码自己进行编译和创建初始化数据库。****
3. **登录账号:目前只能通过已经注册的账号进行登录。**

**三、主界面介绍**

* **好友列表**：显示您的好友列表。
* **聊天窗口**：与单个好友进行聊天的地方，支持发送文字、图片、文件等。

**四、核心功能**

1. **聊天功能**
   * 文本聊天：在聊天窗口输入文字并发送。
   * 表情包：点击聊天窗口中的表情包，选择内容进行发送。
   * 截图：点击聊天窗口中的截图图标，选择区域进行截图。
   * 文件传输：点击聊天窗口中的文件图标，选择文件发送。
2. **好友管理**
   * 添加好友：通过搜索方式添加新好友。

**五、常见问题与解决**

1. **无法登录**：检查账号和密码是否正确，网络是否连接正常等情况。
2. **无法显示**：检查网络、服务器是否正确连接，若有则重启软件进行尝试。

**四、关键技术及难点**

* **注册类**

**使用宏注册，并使用create接口让qml通过自己提供的方法创建对象并注册。**

* **消息传输和信息传输**

**在网络传输中，确保数据完整性和格式正确是至关重要的。我们项目中用的数据包格式，其中包括包头、包长、消息类型、用户ID、数据长度、实际数据内容以及图片数据、包尾。这种消息格式能够有效传递信息，防止数据丢失或者沾包现象的发生。并在接收端使用一个循环缓冲区接受信息，让断包能够暂存并等待新的数据。**

* **多文件传输**

**主要难点是对线程资源的回收，以及文件到消息的映射，为此通过连接信号与槽和设置messageId的解决。除此之外还有对上传文件和下载文件的请求的判断并分别处理，对传输失败文件的删除。**

* **消息解析**

**信息解析包括从输入框到打包，打包到发送，接收到建立缓存，然后回显。在图片和文字混合数据进行粘贴的的时候，首先需要对隐藏文档信息进行解析，分别读出文字信息和不同格式的图片然后对图片进行调整大小和缓存之后再插入非隐藏文档。当点击发送的时候，就会再次解析消息中的图片建立图片List并发送文字和图片。**

**接收到消息的时候，对消息进行解析，对其中的图片进行缓存（如果是离线消息则显示”离线图片”）**

**五、开发心得与体会**

**吴博渊开发心得与体会：**

在利用QML与C++开发仿QQ的数据库与服务器系统过程中，我深刻体会到了跨平台开发的便利性与挑战并存的特性。QML的声明式编程使得UI设计直观且易于维护，能够快速迭代界面，提升用户体验。而C++的强大性能则确保了后台逻辑处理的效率与稳定性，尤其在数据库操作和网络通信方面展现出无可比拟的优势。

开发数据库时，采用SQLite等轻量级数据库，结合C++的Qt SQL模块，有效管理用户数据，保证了数据的完整性和安全性。在服务器开发上，利用Qt Network模块搭建TCP/IP通信，实现了客户端与服务器之间的稳定数据传输。这一过程不仅锻炼了我的网络编程技能，也让我对并发处理、数据同步等高级概念有了更深入的理解。

然而，项目中也遇到了诸多挑战，如多线程管理、数据同步冲突、大规模用户并发处理等。通过不断调试与优化，我学会了如何更有效地利用Qt框架提供的工具与机制来解决这些问题。总的来说，这次开发经历极大地拓宽了我的技术视野，加深了对Qt框架及网络编程的理解，同时也锻炼了解决实际问题的能力。

**邓鑫开发心得与体会：**

在利用QML与C++开发仿QQ的QML前端页面及截图功能我主要使用了QScreen和Qpixmap类，并在深入应用各种信号的过程中，我深刻感受到了QML的灵活性与C++的强大支撑力。QML的声明式UI设计让前端页面的布局与样式调整变得直观且高效，能够快速响应设计需求的变化。同时，QML与C++的无缝集成，使得复杂的逻辑处理如截图功能得以在C++中实现，并通过信号与槽机制与QML前端紧密交互。

信号与槽的使用是Qt框架的一大亮点，在仿QQ项目中，我的截图功能充分利用了这一点，实现了前端与后端的高效通信。无论是用户点击按钮触发的操作，还是数据更新需要通知前端展示，信号都扮演了关键角色，使得代码结构清晰、易于维护。

开发过程中我也遇到了挑战，比如何优化QML页面性能、如何高效处理大量信号等。通过不断学习和实践，我逐渐掌握了相关技巧，提升了开发效率。总的来说，这次开发经历让我对QML与C++的结合应用有了更深刻的理解，也为我未来的Qt项目开发奠定了坚实的基础。

**黄秋汕开发心得与体会：**

在使用QML开发仿QQ的QML前端，我深刻体会到了QML的强大与灵活性。GridView为表情包的展示提供了完美的布局解决方案，使得动态加载与展示大量表情包变得简单高效；SplitView组件可用于创建可调节大小的拆分界面，具有水平和垂直切割的能力，使得界面更加灵活、程序使用更加舒服；简单动画精灵的应用却让界面更加平滑、自然。当然，有些细节的把握真的让人又气又恼，重新优化组件时，出现各种光怪陆离的状况都已是平常。

完善所有客户端页面的过程中，我更加注重用户体验的细节处理，如页面间的流畅过渡、控件的响应速度等。通过不断调试与优化，提升了整个应用的稳定性和性能。

这次开发经历不仅锻炼了我的QML编程技能，也让我更加深入地理解了如何构建高性能、易维护的客户端应用。同时，我也对Qt框架的生态系统有了更全面的认识，为未来的项目开发积累了宝贵经验。